COUNTING THE NUMBER OF ANIMALS OF EACH SPECIES PRESENT IN A SEQUENCE OF IMAGES

Zi Won Cheong, Junhyeong Lee, Kihong Seong
Graduate School of Data Science, Seoul National University, Korea
{jiwon4j ,lrainsoul, kev94yo}@snu.ac.kr

Abstract

We proposed an animal detection and counting model in this paper, and this project belongs to Kaggle competition: iWildcam 2021 - FGVC8. Different from proposed approaches of using maximum computation for counting detected animals by other participants, our model identified each animals independently by adopting centroid tracker which calculates Euclidian Distance between two objects: one from the past frame and the other from the current frame. The motivation of this idea is that counting maximum number of animals in each image cannot capture the whole animals appearing in a series of frames since they may not be in one image. We build our framework by splitting the objective into three main parts: animal detection, animal classification, and animal counting. For each part, we implemented MegaDetector, EfficientNet model, and centroid tracker accordingly. Although we didn't generate the ideas in practice, multiple proposal methods are introduced for each part of the project in this paper such as Assemble-ResNet for animal classification and Unsupervised Deep Tracking (UDT) for animal counting. With test scores generated by Kaggle competition, we compared the results with other participants and analyzed the differences.

1. Introduction

In wildlife research, camera traps are valuable for many ecologists to naturally identify animals' habitats and activity characteristics without harming the natural environment. Advances in object detection algorithms[6, 7, 17] based on Convolutional Neural Network models have freed wildlife researchers from the time-consuming task of classifying the vast amounts of images collected through camera traps while paving the way for efficient research. However, there are still many difficulties with more precise detection due to various environmental characteristics in the wild. Dynamic outdoor environments, such as highly light-free dark environments, fast movements of animals, tiny detection areas due to distance from cameras, poor weather conditions, and obscurities by many obstacles, are significant factors that can degrade object detection algorithms through camera traps. Furthermore, the task of accurately identifying the population of each species caught through camera traps, beyond simply identifying the behavioral characteristics of species through detection, is one of the pretty challenging problems.

While recognizing and classifying animals can be seen as being in line with existing general object detection algorithms, there are areas to be cautious in that the detection target is limited to animals. Moreover, the image can vary dynamically depending on environmental characteristics. Previous studies in the wild environment showed excellent performance[15] in detecting different species, but poor results[2] for new species and environmental differences not experienced in training data. In other words, there was a vulnerable problem in terms of generalization. In addition, there were problems with recognizing obstacles such as rocks as animals and the excessive number of empty images without anything being caught. Beery et al[1] proposed a new framework through mega detectors to overcome these problems and build a more robust model for generalization. Their framework prioritized detection algorithms that identified whether animals were recognized and boxed areas before classifying the exact species and then proceeded with a detailed species classification. It enabled identifying box areas for new species that were previously difficult to detect and quickly filtered unnecessary blank images. Also, through the box area, images could be scaled back to simplify the classification process.

Identifying the number of objects in images or videos[5, 8, 4] is also one of the challenging problems in vision research. The task of identifying the number of animals in camera traps differs from that of identifying the number in a single image in that it identifies the number in a sequence of images based on continuous or irregular time intervals. For example, in a one-minute video, the number of animals detected by each image differentiated over time interval will vary from 0 to 10. It can be accessed simply through

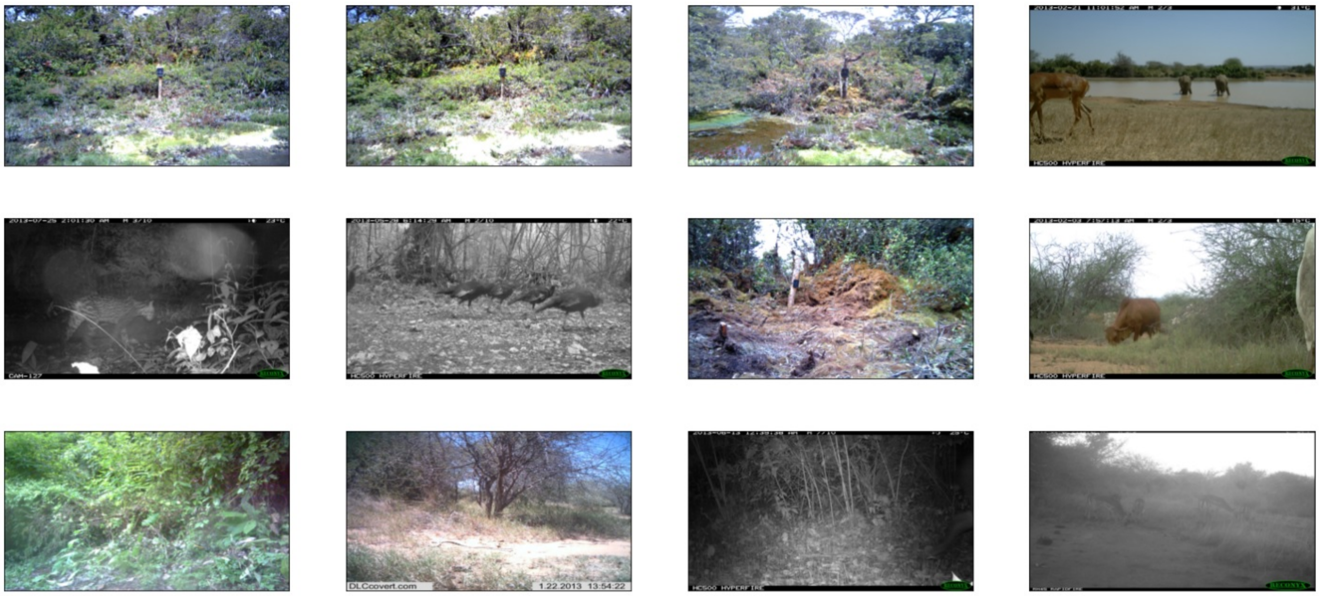

Figure 1. Examples of camera trap images. Note that many camera trap images detect empty images which can produce falsely detected bounding boxes.

a maximum number, but there is a limitation because the maximum may not represent the total number of animals in some cases. Under extreme assumptions, if one unique animal appeared in each of the ten sequence images, the maximum would be 1, but the actual number of animals that appeared would be ten. The unique distinction between the same species of size and appearance in images at different time zones will be challenging. However, with recurrent approaches or object tracking algorithms that can reflect information between images, we can expect better performance than naive approaches. Based on Siamese networks[3], one of the most usual methods of object discrimination, various algorithms[10, 23, 11] have been developed to distinguish objects between different time zones. Moreover, based on sequence data, LSTM methods or Hidden Markov models(HMM)[16] can build improved performance and robust models[9, 22] in object tracking. Furthermore, Yang et al[26] were able to leverage the hybrid data association method to take a global optimization approach to obtain excellent Multi-Object Tracking(MOT) performance.

In order to identify the behavioral characteristics and populations of animals through camera traps, we must be able to recognize and classify animals under various circumstances accurately. Moreover, it is necessary to successfully estimate the number by tracking each object in the image. In addition, considering that it is a camera trap used in wild environment, it is necessary to consider a direction in which computation is relatively light. To this end, we propose the following framework based on the approach of Beery *et al*. [1].

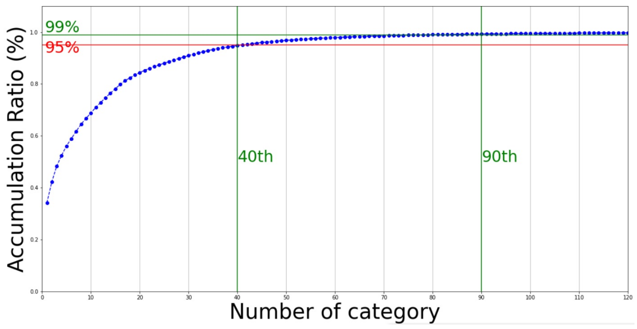

Figure 2. Relationships when cumulative sums are taken in order of increasing categories. It shows that about 40% of the major species occupy 95% of of the data.[14]

1. Based on the Mega detector, we identified whether animals are recognized and derive bounding boxes.
2. We proceed to classify animals using CNN-based classification model. Considering computation, we use cropped images as bounding box regions. Furthermore, we use techniques such as data augmentation and undersampling to prevent bias due to the imbalanced data, considering that most of the major species usually occupy the majority.
3. Finally, we proceed to count the animals in each image sequence. Considering that there is no information about the actual count number, a simple object tracking-based method that does not require train is used.

2. Related Work

2.1. Animal Detection

With given video frames, the first process is to extract bounding boxes that contain objects in an image, and employing MegaDetector V3 can be appropriate since it is a specialized detector tool for wildlife environment which are natural characteristics of our data. Microsoft AI for Earth developed this machine learning model to assist conservation biologists to save unnecessary time looking at series of images to detect animal species manually. In fact, the application of machine learning to camera traps is technically inevitable since manual tracking and detection costs too much time and resources. For instance, false triggers of camera traps produce meaningless images that nearly 70 percent of images are known to be empty.

MegaDetector[1] is composed of two main parts which are generalizable detector and classifier, and the used detector model for this project is based on Faster-RCNN using InceptionResNet network. Despite the ability of the model to create thousands of bounding boxes to capture animal species in an image, it solely categorizes an object between animal and human classes. Using MegaDetector alone is insufficient in terms of classification performance because we have 205 classes in the given data. Therefore, applying an additional ConvNet model to cropped images from MegaDetector is required.

RetinaNet[13] is another modern animal detection model in camera traps. It is an algorithm based on a ResNet model that can lead to similar or better performance than Faster R-CNN, showing similar levels to SSDs in terms of speed. It is commonly used for general object detection tasks and especially for dynamic objects such as automatic vehicles and pedestrian detection. Shepley *et al*. [20] was able to use Retinanet to mitigate one of the biggest challenges in the wild environment, the location invariant problem, to show better performance.

2.2. Animal Classification

EfficientNet model is a state-of-the-art ConvNet model introduced by Google in 2019. Unlike previous ConvNet models which arbitrarily scale network dimensions, such as width, depth, and resolution, EfficientNet [21] uniformly scales each dimension with fixed coefficients. This scaling method is called compound scaling, and its performance shows up to 10 times higher efficiency and accuracy. In other words, EfficientNet improves the accuracy of the model by extending depth, width, and resolution over baseline networks with balanced scales. Especially, EfficientNet-B7 model[21] is known to achieve the highest

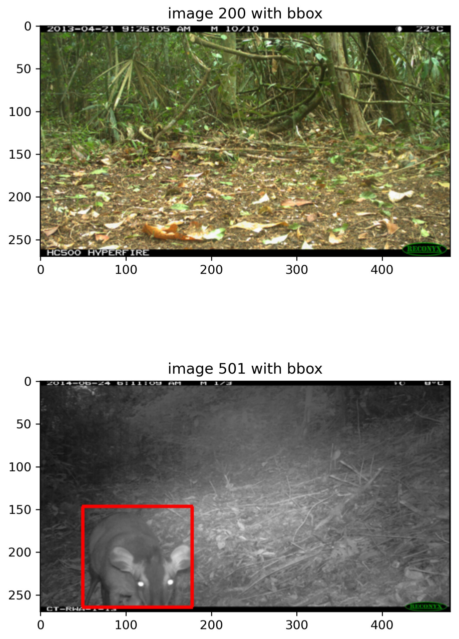

Figure 3. The figure on the top shows no bounding box because an animal is not found in the figure. In fact, numerous images in training data are empty without animals. The figure on the bottom shows that MegaDetector has successfully captured an animal by drawing red bounding box.

accuracy on ImageNet and to be 8.4 times smaller and 6.1 faster than the best existing ConvNet model.

Assemble-ResNet[12] is another state-of-the-art image classification model. Rather than proposing a new architecture, Assembly-ResNet is a method of assembling multiple CNN-related techniques into a single network, showing similar performance and five times faster than EfficientNet.

2.3. Animal Counting

A simple approach to counting objects shown in a video is to get the maximum counts of each detected species in an image. Since many objects appear simultaneously, counting the maximum number may be reasonable in that it counts objects on a single image. However, for videos with multiple images rather than just one image, this is not a good idea. For example, in the case of images of animals passing in one direction, the number of animals in each image changes over time because the camera trap is fixed. To be specific, when two elephants pass through one image and another is later captured, three elephants are actually in the image while using max count results in two, resulting in an error. Therefore, we considered in a more sophisticated way to recognize animals appearing on multiple images

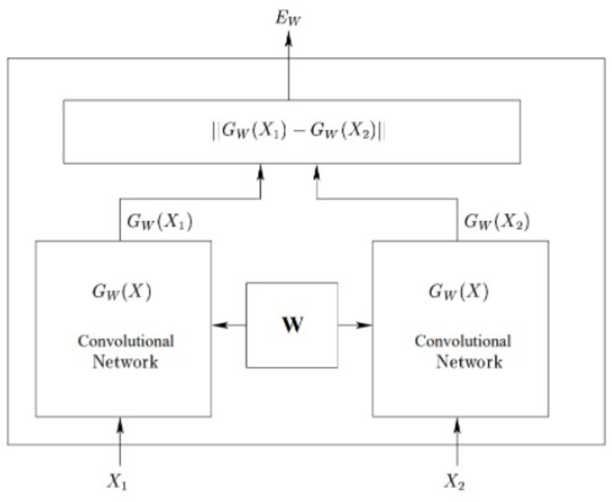

Figure 4. Siamese Network Architecture

and count them.

Centroid tracking is an algorithmic method that allows object tracking relatively simple way without using complex models. It is based on the assumption that the distance of the same objects between frames will be the smallest. First, we compute the centroid of bounding boxes obtained through object detection and give each centroid a unique id. Next, the Euclidian distance between the centroid of the next frame and the previous frame is calculated, and the same id as the previous frame is assigned to the id with the shortest distance. In this process, if an extra centroid occurs, it is assumed that a new object appears. Conversely, you can think of some object as disappeared. Centroid tracking has the advantage of being easy to use without learning, but as the velocity variation of objects increases, the accuracy can be lowered.

$$\text{Centroid Distance} = \|Centroid_1 - Centroid_2\|$$

Another relatively easy-to-use method is **Similarity tracking**. It is identical to centroid tracking in that it distinguishes objects through their distance but differs in that it uses features of objects. Siamese network[3][4] can be used as a representative way to distinguish objects through the distance of features obtained through two convolutional neural networks sharing weights. It is based on the assumption that features obtained through the same convolutional neural network will have similar characteristics. Thus, it has the advantage of being able to approach relatively more precisely than centroid tracking. But the computation is more extensive than centroid tracking, and the performance may be lowered if the object has various characteristics depending on the angle or there are many similar objects.

We can also approach this problem through an unsupervised learning model. Unsupervised object tracking is not a much-researched field yet, but it is a realistically necessary model because there are many cases where there are no true labels in real world. UDT model[24] is an unsupervised deep tracking model based on DCFNet[25] used when there is no true label. In addition to general forward tracking, the UDT model added backward tracking to track objects through comparison in both directions without true label information. Different from algorithmic methods, it can train models and correlation filter is used for tracking. However, depending on the size of the image, the computation can be pretty heavy, so it can be challenging to apply to the camera trap. Therefore, rather than using all frameworks of this model, it will be possible to apply forward-backward and correlation filters selectively.

3. Proposed Method

The goal of this project is composed of two parts: categorization of animal species and counting the number of them in a sequence of images. We conducted this experiment by combining three major tasks: animal detection, animal classification, and animal counting into a pipeline which is shown in 5. MegaDetector locates animals in an image and crops them into bounding boxes, and it demonstrates excellent performance when utilized in real-world data. Unfortunately, the limitations of the model, which distinguishes only two classes between humans and animals, failed to classify images of animals into a total of 205 classes. Therefore, it was inevitable to use state-of-the-art ConvNet models to classify animals in cropped images. Thus, with MegaDetector, we cut all the pictures of animals in frames of original images by bounding box and used them as inputs for EfficientNet models. At this point, the label of each cropped image adopted the annotation provided by the existing data. EfficientNet [21] is a more efficient and superior model than previously introduced ConvNet models by utilizing the compound scaling method. We used the model to classify cropped images from MegaDetector as we are given an annotation in which the correct label is written from the training data.

The learning methods up to here can be called supervised learning models. However, counting animals by species in each image could not use the supervised method because there is no information about counts in training data. Therefore, we computed the count of each animal in an image using a technique that tracks between objects that appear frame by frame using Euclidean distance. Euclidean distance is a very simple and intuitive counting

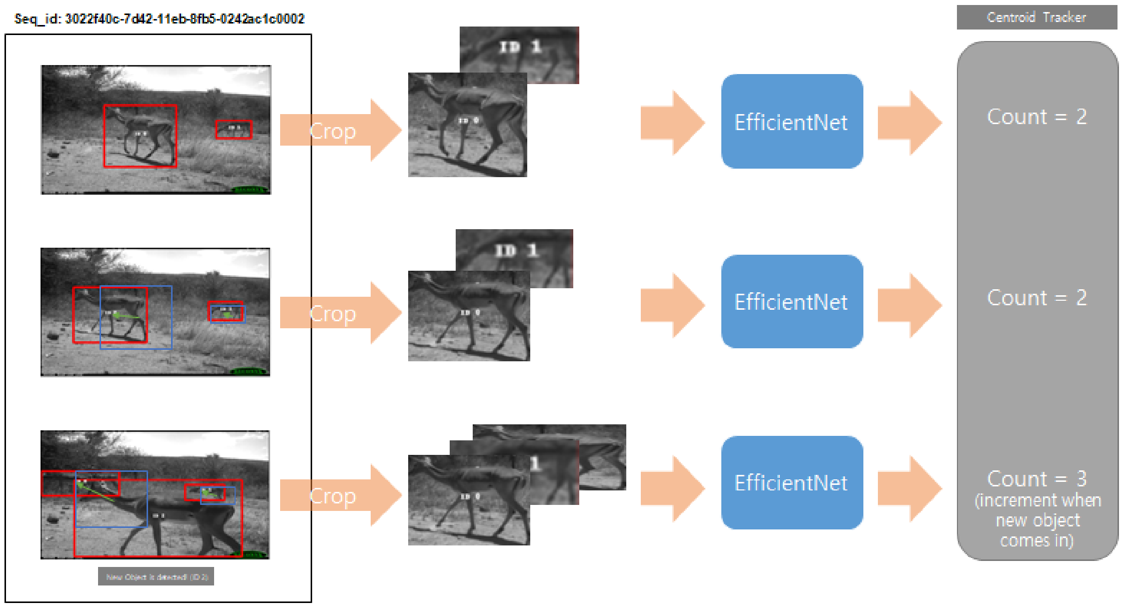
Seq_id: 3022f40c-7d42-11eb-8fb5-0242ac1c0002
Crop
Crop
Crop
New Object is detected! (ID 2)
ID 1
EfficientNet
EfficientNet
EfficientNet
Centroid Tracker
Count = 2
Count = 2
Count = 3
(increment when new object comes in)

Figure 5. The figure shows a pipeline of the proposed model. The first left box represents a video with multiple frames showing that a group of deer passing in left direction. The red bounding boxes indicate the current positions, while the blue bounding boxes show the past position which is in one frame before. Using centroid tracking, we can track independent animals frame by frame. Cropped images in each frame are used as inputs of ConvNet model, then we used a centroid tracker model for counting process.

method. When an animal is placed in a different position in a different frame, tracking an animal by calculating Euclidean distance is a much more logical approach than simply counting the maximum number of animals from one image.

4. Experiment

4.1. Dataset

The dataset is shared by the iWildcam 2021 challenge on Kaggle. As it can be seen from its subtitle, “Count the number of animals of each species present in a sequence of images,” the competition has its main objectives on object detection and counting those objects. To progress the project, both the training data and test data are collected from different camera traps distributed around the globe. The set of animal species shown on different cameras can overlap, but this does not mean they are identical (since cameras are at different locations). The original camera trap data is provided by Wildlife Conservation Society (WCS), with the training set containing 203,314 images from 323 locations, and the test set containing 60,214 images from 91 locations.

Since these images are collected from series of video frames which can be seen in 6, we clustered images by their sequence id written in annotation data representing which video they belong to. Annotation dataset is provided as labels for the WCS dataset which contains various information about the image itself such as images id, sequence id, image size, category, etc. For training purposes, we created additional annotation data to record results of generating animal detection such as bounding box coordinates and classification for each object detected in each image. Bounding boxes in the annotations are generated from MegaDetector V3. Furthermore, we employed an additional Convolution Net model to classify animals into categories. One of the factors that make this project challenging is that given annotation data contains animal species, but not numbers of them. For this reason, generating a supervised learning model to learn and predict the number of animals given to each image became impossible, and a new approach was required to count animals more precisely.

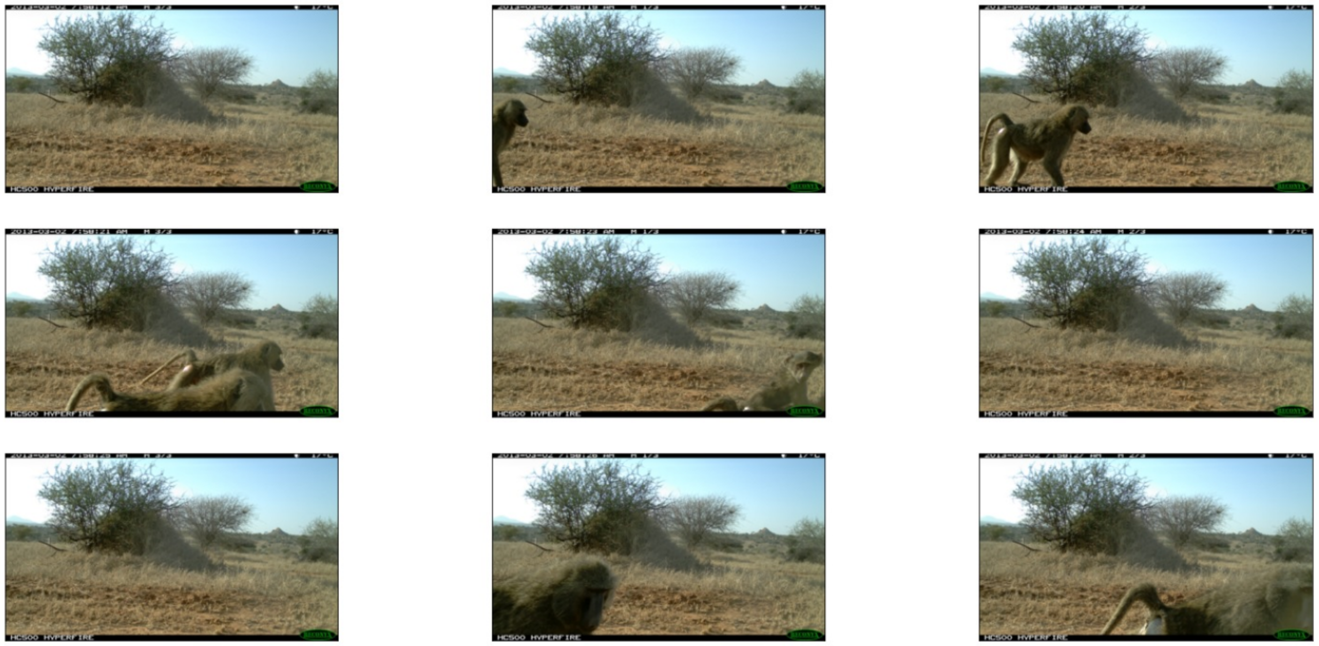

Figure 6. iWildcam image sequences

4.2. Evaluation Metrics

The final result we make is a record of the species of animals in each video and the number of animals of that species. Kaggle evaluates the results we derive with scores computed using Mean Columnwise Root Mean Squared Error (MCRMSE). According to Kaggle competition, they chose this metric to capture both the misidentification of the species and the mistakes in the count, and to ensure that incorrect predictions on empty sequences contribute to the error. The equation for MCRMSE is shown below.

$$MCRMSE = \frac{1}{m}\sum_{j=1}^{m}\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_{ij} - y_{ij})^2}$$

Each column j represents a species, and each row i represents a sequence of images. x_{ij} is the predicted count for that species j in sequence i, and y_{ij} is the corresponding ground truth count. This evaluation of species counts will be done by the iWildcam 2021 competition hosts after submission. In the meantime before final submission, we will evaluate the ConvNet part of the model (EfficientNet) with a simple cross entropy loss for multi-classification.

4.3. Implementation Details

The parameter settings of EfficientNet and the specifications of how it is trained, follow the guidelines from a notebook submitted by username Nayu.T.S.[14]

The code implementations for centroid tracking is based an article by pyimagesearch.[18], and is modified to fit this paper's purpose.

4.3.1 EfficientNet Training

EfficientNet model is initialized from the TensorFlow library, with the model architecture following the EfficientNet-B0 baseline network from the original paper [21]. The network uses 1 initial convolution layer and 7 inverted residual layers also known as MBConv blocks, followed by 1*1 convolution, pooling, and a fully connected layer to produce the final output. MBConv block is an idea from the MobileNetV2 paper [19], and adopts the idea of inverted residuals and bottlenecks. Detailed explanation of MBConv blocks is beyond the scope of this paper, and can be found in the original MobileNetV2 paper. The model's output dimension is set to 205, since we have to classify a given animal image to a total of 205 classes.

To train EfficientNet to classify wild animals, we first crop the detected animals from the given images, using bounding box coordinates given by MegaDetector. Then these cropped images are stored as separate image files, along with labels for each image. After this process is complete, The cropped images are fed into EfficientNet model initialized above, after some data augmentation measures and undersampling methods. Data augmentation is known to help fight overfitting and make the model more robust, so our implementation uses horizontal flip, shift scale rotation, and random brightness contrast as augmentation measures. Only the train data is used for training, and information on the dataset is defined in 4.1. Undersampling seemed necessary since the training data is imbalanced, with most data skewed towards having no animals in it. Therefore, a undersampling method called 'random undersampler' is implemented with the SKlearn library.

After data model and data preparation is ready, we set the training hyperparameters. The model is optimized with Adam optimizer, with a learning rate of 1e-4. Loss function is a simple cross entropy loss which fits our classification task. Batch size is set to 32, and the model is trained for 300 epochs on a single TITAN RTX GPU.

4.3.2 Centroid Tracking

Basic object tracking is done in three steps. First, the tracker takes an initial set of object detections. Second, a unique id is created for each of the initial detections. Third, for the follow up frames in the video, initial objects are tracked and new objects are assigned with unique ids. The object detection results are represented by bounding box coordinates.

Our choice for the algorithm under object tracking is centroid tracking, which is simple but provides adequate performance. It is described in 2 but we will dive deeper into how we implemented the algorithm in this section. Centroid tracking relies on the euclidean distance between already identified object centroids and newly identified object centroids between two back to back frames in a video. The algorithm first calculates the centroid coordinates of existing bounding boxes and new bounding boxes in the form of (x coordinate, y coordinate). The bounding boxes are provided by the competition hosts, and are detected by MegaDetector. After centroid coordinates are calculated, the euclidean distances between new bounding boxes and existing bounding boxes are calculated. Then with these distances, we assume that the same object in subsequent frames will be of closer distance than different objects. With this assumption, the existing objects or bounding boxes are connected to the closest new objects, and the new objects which are not connected to any previous objects after this process are labelled as new objects that comes in, and are assigned unique ids.

To deal with situations when existing objects disappear from the video, we apply the simple assumption that if an existing object does not appear for a number of frames, it has disappeared. We set the boundary to 1 frame. In other words, if an existing object does not appear in the next frame, we deem it as disappeared. This is due to the few frames given on average in the data. Most videos are comprised of less than 5 frames, and it's hard to expect an object to reappear after disappearing in a subsequent frame.

4.3.3 Pipeline

Our models overall pipeline combine EfficientNet and centroid tracking into a single pipeline that takes in a sequence of frames, process each frame at a time to simultaneously track objects and classify new objects. Whenever a new object appears, the classification result for that object produced by EfficientNet is taken into account with the overall class count. Simply put, if a new object is classified as a dog, the count for dog in the overall class count increases by 1. For every frame given, the MegaDetector detection results are used to create bounding boxes for centroid tracking, and the image inside those bounding boxes are cropped and fed into our trained EfficientNet model to produce classification results. This method solves the problem that arises with the max counting method described in 2.3. Whenever a new object appears, the count for the new object's class increments, so for example, when a dog appears in frame 1 and disappears in frame 2, then a new dog appears in frame 3, max counting will count 1 for dogs, but our method using centroid tracking will count 2 for dogs, which is the right answer. After this pipeline goes through, the resulting counts for each video or sequence of test images are aggregated into one csv file, and handed in for submission for the Kaggle competition website.

4.4. Results

Results are shown in Table 1. Unfortunately, our model did not produce better results than the simple max counting method. This is probably due to the fact that most predictions should actually be zero since the original data is extremely sparse. In this case, just filling the whole submission with zeros produces a great result, while slight deviations from the actual ground truth produces worse results. And the second reason is that there was some data with too long intervals between frames, which is thought to have caused problems that made object tracking algorithm meaningless. Also, we found out that some detections by MegaDetector were not adequate that it falsely detect empty objects. We believe that if these faulty detections by MegaDector are improved, our model will reveal its true potential.

We also checked if this was due to non optimized value

Ranking	Team name	Score
1	UFAM	0.02933
2	Sensing Clues	0.02938
3	JuanCarlos	0.03086
4	Charlie Turner	0.03088
5	Yizhen	0.03365
6	xumi	0.03390
7	**GSDS(max bbox)**	**0.03482**
7	**Max counting bbox**	**0.03482**
8	Deva	0.03763
9	**Zero counting**	**0.03840**
10	**GSDS(centroid)**	**0.07222**

Table 1. Kaggle Competition LeaderBoard. Our result(GSDS) is late submission.(not a public ranking)

of the number of frames (k) that determines disappearance of an object. After attempting inference with k ranging from 2 to 10 (up to 10 since 10 is the most number of frames in the dataset), the initial score of 0.7745 when k = 1 decreased to 0.7222 when k = 9. Although there was improvement, it was not of great magnitude, and it is assumed that the second reason mentioned above related with long intervals between frames caused object tracking to be insignificant in this task. If consecutive frames are happening in real time, it is obvious that objects that are in similar positions will be same objects. However, when consecutive frames are far apart in time, it is hard to tell if a completely new object appeared or the same object still exists, just with object tracking.

5. Conclusion

In this work, we introduced an easy and simple framework that counts how many of each species is visible across a sequence of images through camera traps. Unfortunately, the method using centroid tracking did not perform better than simple max counting. It is presumed to mean that there are some cases where the interval between some frames is longer than 10 seconds and that using the tracking method may degrade performance. In the future, we plan to use a method that applies object tracking separately in such cases. Also, we plan to apply Siamese network and unsupervised multi-object tracking methods to animal counting. Furthermore, we will attempt to boost the performance of the pre-trained mega-detector, which can increase the performance of detection at the front of the framework.

References

[1] Sara Beery, Dan Morris, and Siyu Yang. Efficient pipeline for camera trap image review. *arXiv preprint arXiv:1907.06772*, 2019. 1, 2, 3

[2] Sara Beery, Grant Van Horn, and Pietro Perona. Recognition in terra incognita. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 456–473, 2018. 1

[3] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a" siamese" time delay neural network. *Advances in neural information processing systems*, 6:737–744, 1993. 2, 4

[4] Prithvijit Chattopadhyay, Ramakrishna Vedantam, Ramprasaath R Selvaraju, Dhruv Batra, and Devi Parikh. Counting everyday objects in everyday scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1135–1144, 2017. 1

[5] Hisham Cholakkal, Guolei Sun, Fahad Shahbaz Khan, and Ling Shao. Object counting and instance segmentation with image-level supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12397–12405, 2019. 1

[6] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015. 1

[7] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014. 1

[8] Eran Goldman, Roei Herzig, Aviv Eisenschtat, Jacob Goldberger, and Tal Hassner. Precise detection in densely packed scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5227–5236, 2019. 1

[9] Daniel Gordon, Ali Farhadi, and Dieter Fox. Re ˆ3: Re al-time recurrent regression networks for visual tracking of generic objects. *IEEE Robotics and Automation Letters*, 3(2):788–795, 2018. 2

[10] Minyoung Kim, Stefano Alletto, and Luca Rigazio. Similarity mapping with enhanced siamese network for multi-object tracking. *arXiv preprint arXiv:1609.09156*, 2016. 2

[11] Laura Leal-Taixé, Cristian Canton-Ferrer, and Konrad Schindler. Learning by tracking: Siamese cnn for robust target association. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 33–40, 2016. 2

[12] Jungkyu Lee, Taeryun Won, Tae Kwan Lee, Hyemin Lee, Geonmo Gu, and Kiho Hong. Compounding the performance improvements of assembled techniques in a convolutional neural network. *arXiv preprint arXiv:2001.06268*, 2020. 3

[13] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 3

[14] Nayu.T.S. kaggle iwildcam starter notebook, 2021. 2, 6

[15] Mohammad Sadegh Norouzzadeh, Anh Nguyen, Margaret Kosmala, Alexandra Swanson, Meredith S Palmer, Craig Packer, and Jeff Clune. Automatically identifying, counting, and describing wild animals in camera-trap images with deep learning. *Proceedings of the National Academy of Sciences*, 115(25):E5716–E5725, 2018. 1

[16] Lawrence Rabiner and Biinghwang Juang. An introduction to hidden markov models. *ieee assp magazine*, 3(1):4–16, 1986. 2

[17] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *arXiv preprint arXiv:1506.01497*, 2015. 1

[18] Adrian Rosebrock. Centroid tracking, 2018. 6

[19] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks, 2019. 6

[20] Andrew Shepley, Greg Falzon, Paul D Meek, and Paul Kwan. Automated location invariant animal detection in camera trap images using publicly available data sources. *Authorea Preprints*, 2020. 3

[21] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pages 6105–6114. PMLR, 2019. 3, 4, 6

[22] Mohib Ullah and Faouzi Alaya Cheikh. A directed sparse graphical model for multi-target tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1816–1823, 2018. 2

[23] Bing Wang, Li Wang, Bing Shuai, Zhen Zuo, Ting Liu, Kap Luk Chan, and Gang Wang. Joint learning of convolutional neural networks and temporally constrained metrics for tracklet association. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–8, 2016. 2

[24] Ning Wang, Yibing Song, Chao Ma, Wengang Zhou, Wei Liu, and Houqiang Li. Unsupervised deep tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1308–1317, 2019. 4

[25] Qiang Wang, Jin Gao, Junliang Xing, Mengdan Zhang, and Weiming Hu. Dcfnet: Discriminant correlation filters network for visual tracking. *arXiv preprint arXiv:1704.04057*, 2017. 4

[26] Min Yang, Yuwei Wu, and Yunde Jia. A hybrid data association framework for robust online multi-object tracking. *IEEE Transactions on Image Processing*, 26(12):5667–5679, 2017. 2